SESUG 2020 – Paper 148

NHANES Dietary Supplement component: a parallel programming project

Jayanth Iyengar, Data Systems Consultants LLC

ABSTRACT

The National Health and Nutrition Examination Survey (NHANES) contains many sections and components which report on and assess the nation's health status. A team of IT specialists and computer systems analysts handle data processing, quality control, and quality assurance for the survey. The most complex section of NHANES is dietary supplements, from which five publicly released data sets are derived. Because of its complexity, the Dietary Supplements section is assigned to two SAS® programmers who are responsible for completing the project independently. This paper reviews the process for producing the Dietary Supplements section of NHANES, a parallel programming project, conducted by the National Center for Health Statistics, a center of the Centers for Disease Control (CDC).

INTRODUCTION

The National Health and Nutrition Examination Survey (NHANES) is a nationwide health survey which assesses the health and nutritional status of adults and children across the United States. NHANES data are produced by the Division of Health and Nutrition Examination Survey, within the National Center for Health Statistics. The Division of Health and Nutrition Examination Survey is divided up into branches, such as the Operations branch, and the Informatics branch. The Informatics Branch (IB) consists of federal employees, and private contractors who are responsible for loading and processing the data collected for the survey. Specifically, IB consists of IT specialists, computer systems analysts, and SAS programmers who use SAS to do the majority of data processing for the survey. IB staff are also responsible for conducting quality control, and quality assurance for the survey.

BACKGROUND OF THE NHANES SURVEY

The NHANES program started in 1959. Initially, long-term studies were conducted over a period of 4-8 years, and data was released to the public with the same time frequency. Initially, the program was called NHES (National Health Examination Survey). The First National Examination Survey (NHES I) was conducted from 1959-1962, and focused on adults. The Second National Examination Survey (NHES II) was conducted from 1963-65, and focused on children. The Third National Examination Survey (NHES III) was conducted from 1966-1970.

In the 1970's a nutrition component was added, and the survey was renamed The National Health and Nutrition Examination Survey. NHANES I was conducted from 1971-74. NHANES II was conducted from 1976-1980. NHANES III was conducted from 1988-94. In 1999, a continuous NHANES survey was implemented. From that point data was collected and released to the public in 2 year increments.

The data which is released from NHANES is used to determine the prevalence of major diseases, and risk factors associated with the diseases. Information from the survey is used to associate nutritional status with health promotion and disease prevention. The data released is also used in epidemiological and health sciences research to develop and formulate sound public health policy. Another use of the data is to direct and design health programs and services.

The NHANES survey uses a nationally representative sample of approximately 5000 individuals each year. 2 years' worth of data represents 10000 individuals sampled. In order to produce reliable estimates, NHANES oversamples the elderly populations (people aged 60 or older), and minority groups traditionally underrepresented in surveys, including African-Americans, and Latinos.

The survey consists of an examination component, as well as a health interview. The examinations take place in a semi-truck trailer, known as a mobile examination center, or MEC. Mobile examination centers are setup in dozens of locations across the country to examine survey respondents. Within each MEC, is a study team which consists of a physician, medical and health technicians, as well as dietary and health interviewers. During the examination medical, dental, and physiological measurements are taken, as well as laboratory tests administered by highly trained medical personnel.

The NHANES interview includes demographic, socioeconomic, dietary, and health-related questions. Health interviews are conducted in respondents' homes. The interview surveys risk factors, those aspects of a person's lifestyle, constitution, heredity, or environment that may increase the chances of developing a certain disease or condition. The Health interview surveys health behaviors, such as smoking, alcohol consumption, sexual practices, drug use, physical fitness and activity, weight, and dietary intake. Data on certain aspects of reproductive health, such as use of oral contraceptives and breastfeeding practices, are also collected by the interview. The interview also surveys medical conditions and diseases, including cardiovascular disease, diabetes, osteoporosis, respiratory disease, and obesity.

An advanced computer system using high-end servers, desktop PCs, and wide-area networking collect and process all of the NHANES data, nearly eliminating the need for paper forms and manual coding operations. This system allows interviewers to use notebook computers with electronic pens. The staff at the mobile center can automatically transmit data into databases through such devices as digital scales and stadiometers. Touch-sensitive computer screens let respondents enter their own responses to certain sensitive questions in complete privacy. Survey information is available to NCHS staff within 24 hours of collection, which enhances the capability of collecting quality data and increases the speed with which results are released to the public.

STRUCTURE OF NHANES

The NHANES survey is divided up into three primary sections; questionnaire, examination and laboratory. The laboratory and the questionnaire sections are the largest and most extensive sections of the survey. Each section of the survey is then further sub-divided into a series of survey components. A survey component is a specific subject or topical area which data is collected on. Each survey component has its own questionnaire instrument which is used to interview respondents.

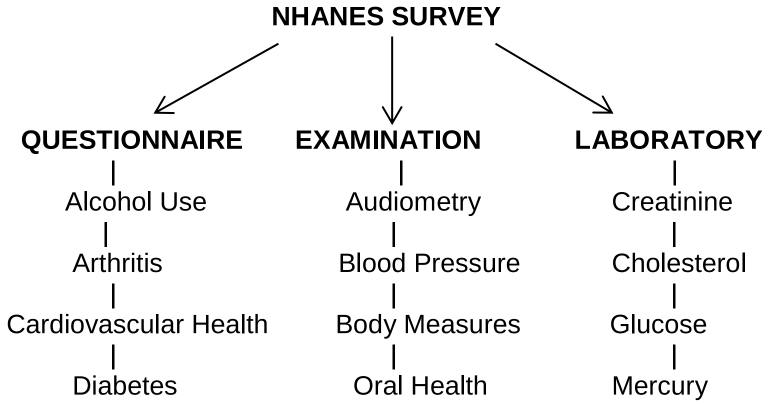

Figure 1. Flowchart – layout of the NHANES survey.

Figure 1 above shows the structure of the NHANES survey with survey sections, and components underneath. In the current release of the survey, the laboratory section contains 64 components, the questionnaire section contains 46 components, and the examination section contains 14 components respectively.

In addition to the three primary sections, there is also a demographics section, which collects and releases data on respondent demographics. The information collected in the demographic section includes race and ethnicity, country of origin, age, gender, and educational status. The demographics section contains a single component.

The Dietary Supplements component of the questionnaire section has long been the most complicated component of the questionnaire section. Because of its size, in the most recent releases of the survey, Dietary Supplements has been given its own section of the survey, and is no longer a component of the questionnaire section.

Starting in 1999, when the continuous survey was launched, the NHANES survey was conducted in two-year survey cycles. Specifically, this means that data was collected and released from the survey every two years. For example, the first survey cycle was 1999-2000, the second was 2001-2002, the third 2003-2004, and so on, and so on.

NHANES DATA PRODUCTION

The process of taking the raw survey data which is loaded to the NHANES server, and converting it to public-use files (PUF's) released to the general public, is known as data production. The process of data production is managed and coordinated by a project manager, and programming staff. The project manager, also known as a project officer, is responsible for several components of each section of the survey. Within the division of NHANES, each section of the survey (questionnaire, examination, laboratory) has its own team which is composed of a team leader, and IT or programming staff, which can be federal employees, or contractors.

For each component of the survey, the team leader assigns the responsibility of data production to a programmer who's a member of the respective team. The programmer is responsible for producing the data files and codebook for a specific component. The final data files are SAS transport files which when validated will be posted to the NHANES website for download by the public. The codebook is essentially a data dictionary. Appendix I contains an excerpt of the codebook from the Dietary Supplements component.

In Figure 2 below is a project management flowchart which shows the different roles and responsibilities involved in a data production project. In addition to the programmer who's assigned to the project, each survey component is also assigned two quality assurance reviewers who validate the data files and codebook. The QA reviewers are also programmers and IT specialists who are members of the respective team. They develop a QA report detailing any corrections to be made to the data files and codebook.

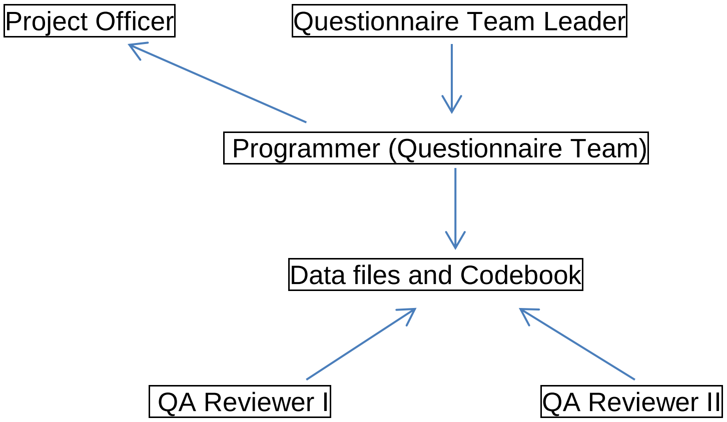

Figure 2. NHANES Data Production Flowchart

The programmer works closely with the project officer to complete the production of the data files. The project officer develops a written specification form for the programmer to follow. The specifications include a list of all the variables to be contained in the final data set, and SAS variable labels for each variable. They also include specific instructions for recoding or making edits or corrections to data, deriving new variables on the final data set, subsetting the data, and naming conventions to use for each variable on the final data set.

Using the specifications, the programmer creates an individual SAS program to implement all the edits, checks, corrections, and derivations to the data required by the specifications. Its common for the specifications to be vague, and also for the project officer to make mistakes in writing up the specifications. Thus, the programmer must work closely with the project officer to confirm specifications, and ensure that they're interpreted correctly.

THE DIETARY SUPPLEMENTS COMPONENT

The Dietary Supplement component of the questionnaire section of NHANES collects information on prescription and non-prescription dietary supplements, non-prescription antacids, prescription medication, and asthma medication. Survey respondents are asked about their use of dietary supplements in a 30-day time period prior to the date of interview. NHANES maintains a product label database which contains product level information that's used to fill in the gaps when respondents report a supplement with incomplete information.

The Dietary Supplements component is the most lengthy and complex component within the questionnaire section, and across the NHANES survey as a whole. The majority of components within the questionnaire section of NHANES involve the production of a single final data set and corresponding codebook as public release files. The Dietary Supplements component requires the production of 5 final data sets and corresponding codebooks.

The five public release files provide information and data on respondents' individual dietary supplement use, total dietary supplement use, product level information, supplement ingredient information, and information on supplements categorized as blends. In Figure 3 below, a table includes the data sets and descriptions for the 5 public release files which constitute the Dietary Supplements component.

Description	Data set
Dietary Supplement Use – Individual Dietary Supplement Use	DSQIDS_F.XPT
Dietary Supplement Use – Total Dietary Supplement Use	DSQTOT_F.XPT
Dietary Supplement Database – Product Information	DSPI.XPT
Dietary Supplement Database – Blend Information	DSBI.XPT
Dietary Supplement Database – Ingredient Information	DSII.XPT

Figure 3. Dietary Supplements component public-release data files

Due to its size and complexity, a separate data production process has been devised for Dietary Supplements. Instead of the component being assigned to a single programmer whose responsible for producing the data files and codebook. The Dietary Supplements is assigned to two programmers who work independently to produce the data files and codebook. This structure is known as parallel programming, or in some environments, double independent programming.

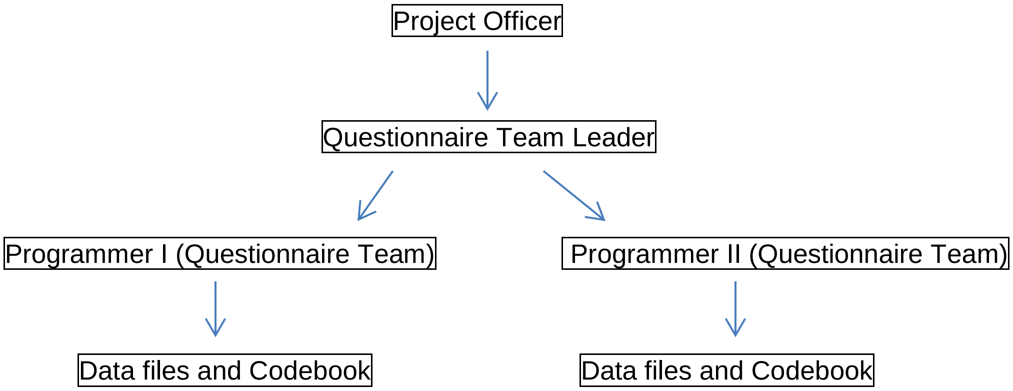
Project Officer
Questionnaire Team Leader
Programmer I (Questionnaire Team)
Programmer II (Questionnaire Team)
Data files and Codebook
Data files and Codebook

Figure 4. Dietary Supplements Data Production and Project Management

Figure 4 above displays the parallel programming process which is utilized for the data production of the Dietary Supplements component.

As a project management structure, parallel programming has been adopted for Dietary Supplements because the project is very lengthy and complex, and requires an extra layer of validation. In general however, it can be argued that parallel programming is inefficient and a waste of resources, i.e., having two people doing the exact same job. In programming projects in most organization environments, parallel programming is only used in rare circumstances, for the most complex of projects.

In the following section we detail the heavy amount of data manipulation involved in the Dietary Supplements component. As we'll see, the high volume of edits involved in the production of the data files for Dietary Supplements further complicates producing accurate, high-quality data sets. With parallel programming, the final data sets produced by the programmers must match. Arriving at this outcome given the amount of edits to the data which Dietary Supplements requires can be quite a challenge.

INDIVIDUAL DIETARY SUPPLEMENT USE

Each of the programmers writes a single unit SAS program to produce the final data sets. The SAS program to produce the individual dietary supplement use final data set is very lengthy, and may include thousands of lines of code because it must include and apply a long list of edits to the survey data. To perform the changes and manipulations, the SAS programs make heavy use of DATA STEP programming.

```
Data DSQ_RCBC;
      *Remove all Records with codes 7-10 and not in Final Sample;
  Set DSQV4 (Rename=(MATCH_CO=DSDMTCH DSQ103=DSD103 DSQ123Q=DSD122Q DSQ123U=DSD122U
                  DSQ071=DSD070) Where=(SEQN^=. and DSDMTCH NOT IN (7 8 9 10)) );

  If DSDSUPID='6666666118' Then DSDSUPP='7777';
  If DSDSUPID='6666666064' Then DSDSUPP='9999';

  If SEQN='47614' and DSDSUPID='1000316701' Then
     DSDSUPP='SUNDOWN Q-SORB CO Q-10 30 MG EASY-TO-SWALLOW';

  If DSDSUPP='7777' Then DSDMTCH=7;
  If DSDSUPP='9999' Then DSDMTCH=9;

  If DSQ096U=1 Then DSD090=DSQ096Q;
  Else If DSQ096U=2 Then DSD090=DSQ096Q*7;
  Else If DSQ096U=3 Then DSD090=DSQ096Q*30.4;
  Else If DSQ096U=4 Then DSD090=DSQ096Q*365;
  Else If DSQ096Q=9999999 Then DSD090=9999999;
  Else DSD090=DSQ096U;

 If SEQN=64782 and DSDSUPID='1000721800' Then Do; DSD122Q=2; DSD122U=27; End;
 If (SEQN IN (51090 51034) and DSDSUPID='1000778800') or (SEQN=50101 and
     DSDSUPID='1000294300') Then DSD122U=1;

 If (DSDSUPID='1000563800' and DSD103=12) or (DSDSUPID='1000033801' and DSD070=2)
  or (DSDSUPID='1000788900' and DSD103=.) or (DSDSUPID='1888788800' and INSTANCE=4)
  or (DSDSUPID='1000790600' and SDASTAND=238) Then Delete;

Run;
```

Figure 5. Sample SAS Data Step Program – Individual Dietary Supplements Use

In Figure 5 above is an excerpt from the SAS program which produces public release data files for Individual Dietary Supplement Use.

As you can see from Figure 5, the typical SAS program includes a large volume of edits and modifications to the data, which have all been documented in written specifications developed by the project officer. The edits include recodes, backcodes for existing variables, and computations of new variables based on existing ones, as well as other edits.

SEQN or SP_ID is the variable for the respondent. At the top of the program, the WHERE= data set option is used to exclude records missing respondent identifier (SEQN) which are not in the final sample, and records which contain specific values for DSDMTCH. The RENAME= data set option is used extensively for the purpose of renaming existing variables to conform to specified variable names in the specifications.

SUPPID is the dietary supplement identifier. Questionnaire variables must be recoded for specific supplements (SUPPID) reported by individual respondents (SEQN). In some cases, dietary supplement name must be corrected based on supplement identifier. For individual supplements and respondents, values are stipulated for quantity of the supplement consumed daily, and for dosage form of the supplement. To apply all these edits, the SAS program makes heavy use of IF-THEN-ELSE conditional logic.

The DATA STEP assignment statement is used to compute or derive new variables which are listed in the specifications. At the bottom of the step, records for specific supplements in combination with other variables need to be deleted. The IF-THEN-DELETE construct is used here to perform this task.

```
      /* Attachment 1b: DSDDAY1,DSDDAY2 */
      /* Append THREE Core Files to get variables RXQ220 and DSQ130 */

Data DRX_A_E (Drop=DSQLOC);
  Set DSAN.DSDRX_E (Keep=SEQN DSQLOC DSQRCL DSQ130 DSQ052 DSQ049 SP_ID)
      DSAN.DSDRXA_E (Keep=SEQN RXQLOC RXQRCL RXQ220 RXQ141 SP_ID
                       Rename=(RXQLOC=DSQLOC RXQRCL=DSQRCL RXQ220=DSQ130))
      DSAN.ANDRXA_E (Keep=SEQN RXQLOC RXQRCL RXQ220 RXQ141 SP_ID
                       Rename=(RXQLOC=DSQLOC RXQRCL=DSQRCL RXQ220=DSQ130));

   If DSQRCL in (1 2) and DSQ130 IN (. 2 7 9);
Run;

            /* Replace SEQN Missing values. Join with Crosswalk File */
Proc Sql;
 Create Table DRX_XW as
  Select A.*, B.Seqn
 From DRX_A_E(Drop=Seqn) as A, SAMPLE_E.NHXWALK2007_2008 as B
 Where A.SP_ID=B.SP_ID;
Run;
Quit;

Data DRX_D1 DRX_D2;
  Set DRX_XW;

   If DSQRCL=1 and DSQ130 IN (. 2 7 9) Then Output DRX_D1;
   If DSQRCL=2 and DSQ130 IN (. 2 7 9) Then Output DRX_D2;
Run;

Proc Sort Data=DSQV5_ Out=DSQT1; By SEQN DSDSUPID; Run;

Proc Sort Data=DSAN.DS1IDS_E Out=DS1IDS_E Nodupkey; By SEQN DSDSUPID;
Proc Sort Data=DSAN.DS2IDS_E Out=DS2IDS_E Nodupkey; By SEQN DSDSUPID;
Run;

Data T1 (RENAME=(DSQ52=DSQ052 RX141=RXQ141));
  Merge DSQT1(IN=A) DS1IDS_E(KEEP=SEQN DSDSUPID IN=B) DS2IDS_E(KEEP=SEQN DSDSUPID IN=C);

   By SEQN DSDSUPID;

  DSQ52=UPCASE(DSQ052);
  RX141=UPCASE(RXQ141);

  If A and B Then DSDDAY1=1; Else DSDDAY1=.;
  If A and C Then DSDDAY2=1; Else DSDDAY2=.;

  Drop DSQ052 RXQ141;
  If A;
Run;

Proc Freq Data=T1;
    Tables DSDDAY1 DSDDAY2 / LIST MISSING;
Run;
```

```
Proc Sql;
 Create Table T2 as
  Select A.*, B.DSQ130 as DSQ130_D1, B.DSQRCL
  From T1 as A Left Join DRX_D1 as B
  On A.SEQN=B.SEQN and A.DSQ052=B.DSQ052 and A.DSQ049=B.DSQ049 and A.RXQ141=B.RXQ141
  Order By SEQN, DSQ052, DSQ049, RXQ141;
Quit;

Data DSQ_D1;
  Set T2;
  If DSDDAY1^=1 AND DSQRCL=1 Then DSDDAY1=DSQ130_D1;
  DROP SDASTAND SERVDUMV DSQRCL DSQ110 DSQ130_D1 RXQ195O RXQ231 RXQ240B RXQ240G
  RXQ240S RXQ290;
Run;

Proc Sort Data=DRX_D2 Nodupkey;
    By SEQN DSQ052 DSQ049 RXQ141 DSQRCL;
Run;

Data T3(Drop=SP_ID);
 Merge DSQ_D1(IN=A)
       DRX_D2(Rename=(DSQ130=DSQ130_D2) IN=B);
   By SEQN DSQ052 DSQ049 RXQ141;

If A;
Run;
       /*** T1-> T2-> T3 ----->  DSQV6***/
Data DSQV6;
  Set T3;
      If DSDDAY2^=1 and DSQRCL=2 Then DSDDAY2=DSQ130_D2;
   DROP DSQRCL DSQ049 DSQ052 DSQ130_D2;
Run;
```

Figure 6. Individual Dietary Supplement Use – SAS code to derive Dietary Recall variables.

Figure 6 contains the SAS code to derive the dietary recall variables (DSDDAY1, DSDDAY2) from the same SAS program for Individual Dietary Supplement Use. This lengthy section of the SAS process contains an extensive amount of data manipulation and DATA STEP programming.

At the top of the program, three core external files must be combined vertically. The DATA STEP SET statement is used to concatenate and append the three files together. To append input data sets using the DATA STEP, the data sets must contain the same set of variables for the purposes of data validity. The KEEP= and RENAME= data set options are used to ensure the three data sets contains the same set of variables, and that variable names are consistent. The subsetting IF statement is used to subset the data to include only observations which are necessary in the output data set.

In the next step, the SEQN variable is extracted from the crosswalk file. A PROC SQL INNER JOIN is used to combine with the crosswalk file and pull over the SEQN variable. Later in the code, to compute the dietary recall variables, DSDDAY1, and DSDDAY2, a DATA STEP MERGE of three data sets is performed. To prepare for the merge, PROC SORT is run on each of the input data sets to remove duplicate observations based on the combination of BY variables.

Computing the dietary recall variables requires a match to be found between the questionnaire file (DSQT1) and one of the two 24-hour dietary recall files (DS1IDS_E) and (DS2IDS_E), based on supplement reported by a particular respondent. The temporary IN= variables permit the programmer to sort out matches from non-matches. Thus, the DATA STEP merge is a perfect fit for the requirements demanded by this task.

REASONS FOR TAKING A DIETARY SUPPLEMENT

The SAS code for creating the reasons for taking dietary supplement variables is included in Appendix II. The final data set contains 35 variables, with each variable specifying a different reason for taking a dietary supplement. The data for reasons for taking a supplement was collected apart from the main questionnaire interview, and is stored in an external spreadsheet file which needs to be integrated with the questionnaire data.

This section of the Individual Dietary Supplement Use code also places heavy reliance on DATA STEP programming to perform quality control, and quality assurance. The code makes use of some effective SAS programming constructs to perform some data cleaning.

The data contains multiple records where the same respondent (SEQN) reported taking the same supplement (DSDSUPID). PROC SORT with the NODUPKEY option is used extensively to eliminate duplicate records of the same combination of respondent and supplement.

The data needs to be restructured so that data for each specific reason is aligned properly in a SAS variable, DSQ128A – DSQ128II. All the variable names are values contained in the backcode variable, with one record per unique value. Here I used PROC TRANSPOSE to rotate the data so that observations become variables.

I used a DATA STEP MERGE to integrate the reasons data, with the main dietary supplement questionnaire data set. Further, I used arrays along with an iterative do-loop to recode reasons variables with missing values and replace them with valid values from the external reasons data set.

INGREDIENT VARIABLES

The public use data set for Individual Dietary Supplement Use contains 33 variables which report on ingredient levels for each ingredient found within dietary supplements reported consumed by individual respondents. Ingredients of a supplement range anywhere from carbohydrates and fats, to vitamins and minerals, to other nutrients. In Appendix III is the section of the SAS program which calculates ingredient variables.

The SAS code in this section further illustrates the volume of data manipulation required and the complexity involved in deriving the final data set for Individual Dietary Supplement Use. Calculation of the ingredient variables is a multi-step process. First, the public release data set on ingredient information (DSII) is referenced and is used to bring in only supplements and ingredients included in the specification form.

The next step uses an ingredient identifier (DSDINGID) to exclude ingredients for specific supplements not used in calculations. This step makes ample use of the IF-THEN-DELETE construct.

In the ensuing step, quantities and amounts of ingredients levels are set and calculated for specific ingredient and supplement combinations. The code in this step makes tremendous use of IF-THEN-ELSE conditional logic and DO-END groups. PROC TRANSPOSE is used to restructure the data to create variables for each specific ingredient. In the final step, ingredient levels are computed for each variable using actual serving size (DSDACTSS).

THE PITFALLS AND NUANCES OF PARALLEL PROGRAMMING

As discussed earlier in the paper, data production for the Dietary Supplements component is a parallel programming project where responsibility for production of the data files and codebook is assigned to two programmers who write and develop SAS code from scratch, independently of each other.

For the Individual Dietary Supplements use data files, once the programmer is finished with writing and developing, testing, and debugging the code, then it's necessary to execute the code and produce the final results. As a parallel programming project, the two programmers assigned to the component have to compare their results, and determine if any difference exist.

To complete this step, a simple PROC COMPARE of the two final data sets is run. If the final data sets match, then the data has been validated, and the project is complete. However, if there are discrepancies between the two data sets or codebooks, then the two programmers need to work together to resolve the discrepancies.

PROC COMPARE produces multiple sections of output and data set comparisons. It produces a variable summary, a values summary and an observations summary. First, we need to determine whether the data sets contain the same number of variables, and number of observations. This is the first comparison produced in PROC COMPARE output.

One data set may have a different set of variables than the other. The Individual Dietary Supplement Use data set contains 83 variables total. It's possible that one data set may have omitted a variable, or a variable is missing from one data set. A discrepancy in the number of observations may indicate that one of the programmers incorrectly merged or subsetted their data in one section of the code.

Some discrepancies are easier to fix than others. The easiest discrepancy to fix is a discrepancy in variable labels, where there's a different label for the same variable. A discrepancy in variables, where one data set contains more variables than the other, or one data set's missing variables is not too hard to resolve. However, a discrepancy in values, where the same variable contains different values for a given observation, can be quite challenging. Likewise, discrepancies in observations can be difficult to resolve.

In values or observations discrepancies, the issue at hand is determining which data set is right and which is wrong. In other words, ascertaining which programmer is at fault and made a mistake. To do this, each programmer needs to carefully review their code, and take responsibility for their work. Some programmers may not be inclined to do this. They may be reluctant to admit their mistakes, or even cover them up.

What is also at hand is locating the source of the discrepancy. This is a very arduous task in the Dietary Supplement code, since we're dealing with thousands of lines of SAS code where there's a large volume of manipulations, edits and changes being made to the data.

Locating the discrepancy may also involve reviewing, debugging and running the other programmers' code. This can add another level of challenge to the task. The other programmers' code may not be well documented, and legible. It may not incorporate line spacing, indentation, case\capitalization and other elements of style which makes it easier to follow.

Taking note of the multiple sections, length and complexity of the Individual Dietary Supplement Use code in the figures and the appendix, gives us a more precise idea of what's involved in tracking a discrepancy. The source of the discrepancy may stem from the recodes and back codes section, the dietary recall variables, reasons for taking a dietary supplement, or the ingredient variables.

CONCLUSION

This paper introduced and examined the process involved for producing public-release data sets for the Dietary Supplements component of the National Health and Nutrition Examination Survey (NHANES). With 5 public-release data sets, Dietary Supplements is the most complex section of the survey, requiring parallel programming. Producing the final data sets for Individual Dietary Supplement use requires extensive data manipulation and data step programming. Parallel programming, although necessary to validate results, is arduous when discrepancies are found, because of the large volume of manipulations in the Dietary Supplements component. Programmers are required to sift through hundreds of lines of code to find the source of differences.

REFERENCES

Centers for Disease Control and Prevention, National Center for Health Statistics. “National Health and Nutrition Examination Survey, Questionnaires data sets and related documentation”. Accessed 04/16/2020. URL-https://www.cdc.gov/nchs/nhanes/Search/ DataPage.aspx? Component= Dietary & CycleBeginYear =2009

Centers for Disease Control and Prevention, National Center for Health Statistics, National Health and Nutrition Examination Survey, Questionnaires data sets and related documentation, NHANES 2009-10 Dietary Data. “Dietary Supplements Use 30-Day – Individual Dietary Supplements (DSQIDS_F)”. Accessed 04/16/2020. https://wwwn.cdc.gov/Nchs/Nhanes/2009-2010/DSQIDS_F.htm

Iyengar, Jayanth.. October 2011, “Can you decipher the code? If you can maybe you can break it”. Proceedings of the Southeast SAS Users Group 2011 Conference, Alexandria, VA. Southeast SAS Users Group. Available at https://lexjansen.com/cgi-bin/xsl_transform.php?x=sesug2011 and https://analytics.ncsu.edu/ sesug/ 2011/ CC21.Iyengar.pdf

ACKNOWLEDGMENTS

The author would like to thank Richann Watson SESUG 2020 Academic Chair, Lesa Caves, SESUG 2020 Operations Chair, and Rachel Straney and Deanna Schreiber-Gregory, Industry Solutions Section Co-chairs, and the SESUG Executive Committee and Conference Team for accepting my abstract and paper and for organizing the conference.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the author at:

Jay Iyengar
Data Systems Consultants LLC
datasyscon@gmail.com
https://www.linkedin.com/in/datasysconsult/

Jay Iyengar is principal of Data Systems Consultants LLC. He is a SAS Consultant, Trainer, and SAS Certified Advanced Programmer. He is co-leader of the Chicago SAS Users Group, WCSUG. He’s presented papers at SAS Global Forum (SGF), Midwest SAS Users Group (MWSUG), Wisconsin Illinois SAS Users Group (WIILSU), Northeast SAS Users Group (NESUG), and Southeast SAS Users Group (SESUG) conferences. He has been using SAS since 1997. His industry experience includes Healthcare, Pharmaceutical, Public Health, Database Marketing Educational Testing, and International Trade.

TRADEMARK CITATION

DSQICARB - Carbohydrate (gm)

Variable Name:
DSQICARB

SAS Label:
Carbohydrate (gm)

English Text:
Carbohydrate (gm)

Target:
Both males and females 0 YEARS - 150 YEARS

Code or Value	Value Description	Count	Cumulative	Skip to Item
0.17 to 66	Range of Values	1337	1337	
.	Missing	7063	8400	

DSQISUGR - Total sugars (gm)

Variable Name:
DSQISUGR

SAS Label:
Total sugars (gm)

English Text:
Total sugars (gm)

Target:
Both males and females 0 YEARS - 150 YEARS

Code or Value	Value Description	Count	Cumulative	Skip to Item
0.33 to 66	Range of Values	1226	1226	
.	Missing	7174	8400	

```
            /**** Bring in and Edit Backcoding File from spreadsheet ****/
                  /* HH_REASONS and RX_REASONS  */
Data HH_RX_REASON;
  Set DSAN.HH_RX_REASONS(OBS=787);

  If Backcode=' ' Then Do;
      If Code=91 Then Backcode='DSQ128S_';
      If Code=99 Then Backcode='DSQ128A_';

      If Code=. Then Do;
        Backcode='DSQ128S_';
        Code=91;
      End;
  End;
            * SPs without any backcoding instructions\values;
       If Seqn=52100 and DUMPS_ID='1888232001' Then Delete;
       If Seqn=49598 and DUMPS_ID='1888004301' Then Delete;

  Drop SP_ID DSQ128S DSQ128A_;
Run;

                  /* Delete Duplicate combinations of SEQN and Product ID */
Proc Sort Data=HH_RX_REASON Nodupkey;
  By SEQN DUMPS_ID;
Run;

                  /* Transpose Codes to Obtain Variables DSQ128A-DSQ128II */
Proc Transpose Data=HH_RX_REASON Out=HH_RX_TRSP(Drop=_NAME_);
  Var CODE;
  By SEQN DUMPS_ID;
  ID BACKCODE;
  *Copy DSQ129O;
Run;

Data HHRX_TRSP;
  Set HH_RX_TRSP(Rename=(DUMPS_ID=DSDSUPID));8

 If SEQN=42307 and DSDSUPID='1888695801' Then DSQ128J_=19;
 If SEQN=43186 and DSDSUPID In ('1888000500' '1888007900') Then DSD128EE=39;
 If SEQN=51054 and DSDSUPID In ('1888000500','1888753100') Then DSD128EE=39;
 If SEQN=50739 Then DSDSUPID='1888028905';

Run;

Proc Sort Data=DSQV6;
 By SEQN DSDSUPID;
Run;

      /*** DSQV6 ----->  DSQV7 ***/
            /* Backcode Reasons for Supplements from Edit Spreadsheet */
Data DSQV7;
 Merge DSQV6(IN=A) HHRX_TRSP(IN=B);
   By SEQN DSDSUPID;
 If A;
Run;
```

```
      /*** DSQV7 ----->  DSQV8 ***/
Data DSQV8;
    Set DSQV7;

      Array HHRX {19} DSQ128A DSQ128B DSQ128C DSQ128D DSQ128E DSQ128F DSQ128G
      DSQ128H DSQ128I DSQ128J DSQ128K DSQ128L DSQ128M DSQ128N DSQ128O DSQ128P
      DSQ128Q DSQ128R DSQ128S;

      Array NEWHHRX {19} DSQ128A_ DSQ128B_ DSQ128C_ DSQ128D_ DSQ128E_
      DSQ128F_ DSQ128G_ DSQ128H_ DSQ128I_ DSQ128J_ DSQ128K_ DSQ128L_ DSQ128M_
      DSQ128N_ DSQ128O_ DSQ128P_ DSQ128Q_ DSQ128R_ DSQ128S_;

         Do I=1 to 19;
             If HHRX(I)=. and NEWHHRX(I) NE . Then HHRX(I)=NEWHHRX(I);
         End;
Run;

      /*** DSQV8 ----->  DSQV9 ***/
Data DSQV9;
   Set DSQV8;

  Array SUPP_REAS {35} DSQ128A_ DSQ128B_ DSQ128C_ DSQ128D_ DSQ128E_ DSQ128F_
                       DSQ128G_ DSQ128H_ DSQ128I_  DSQ128J_ DSQ128K_ DSQ128L_
                       DSQ128M_ DSQ128N_ DSQ128O_ DSQ128P_ DSQ128Q_ DSQ128R_
                       DSQ128S_ DSD128T DSD128U DSD128V DSD128W DSD128X
                  DSD128Y
                       DSD128Z DSD128AA DSD128BB DSD128CC DSD128DD DSD128EE
                       DSD128FF DSD128GG DSD128HH DSD128II;

      If _LABEL_='Code' Then Do;
            Do i=1 to 35;
                      If SUPP_REAS{I}^=. Then Code=SUPP_REAS(I);
            End;
      End;

      If 10<=Code<=77 Then DSQ128S=.;

      Drop DSQ128A_ DSQ128B_ DSQ128C_ DSQ128D_ DSQ128E_ DSQ128F_ DSQ128G_
      DSQ128H_
           DSQ128I_ DSQ128J_ DSQ128K_ DSQ128L_ DSQ128M_ DSQ128N_ DSQ128O_
           DSQ128P_ DSQ128Q_ DSQ128R_ DSQ128S_  CODE I;
Run;
```

```
               /* Bring in only Supplements\Ingredients specified in ESF  */
               /* FILE: DSQ4_E and INGRID                            */
Proc Sql;
 Create Table TEMP4 as
 Select  Left(Put(INGID, 8.)) as DINGID, UPCASE(LABEL) as DSDINGR
 From DSAN.INGRID;

 Create Table DSUPP_INGRID as
  Select A.DSDSUPID, A.DSDINGID, A.DSDQTY, A.DSDUNIT, B.DSDINGR
   From DSAN.DSQ4_E as A, TEMP4 AS B
  Where A.DSDINGID=B.DINGID;
Quit;

 /* Exclude Ingredients for certain Supplements, not used in Calculations */
Data TEMP5;
    Set DSUPP_INGRID;

               /* Dietary Fiber                                      */
  If DSDINGID='10001179' and DSDSUPID^='1000057400' Then Delete;

  If DSDINGID='10000501' AND
     DSDSUPID IN ('1000414000' '1000057700' '1000057300' '1000057400')
     Then Delete;

            /* Lutein                                          */
  If DSDINGID='10003229' AND DSDSUPID='1000427900' Then Delete;

            /* Niacin                                          */
  If DSDINGID='10000908' and DSDSUPID^='1000146200' Then Delete;
  If DSDINGID='10001089' and DSDSUPID^='1000458400' Then Delete;

            /* Magnesium                                       */
  If DSDINGID='10001656' and DSDSUPID^='1000753700' Then Delete;

            /* Riboflavin                                      */
  If DSDINGID='10001243' and DSDSUPID^='1000438000' Then Delete;

            /* Calcium Edits                                   */
  If (DSDINGID='10002190' and DSDSUPID^='1000330700') or
     (DSDINGID='10000584' and DSDSUPID^='1000317800') or
     (DSDINGID='10002193' and DSDSUPID^='1000330800') or
     (DSDINGID='10000795' and DSDSUPID^='1000671600') or
     (DSDINGID='10004704' and DSDSUPID^='1000589000') Then Delete;

  If DSDINGID='10001144' and
     DSDSUPID Not In('1000269000' '1000269001' '1000320400') Then Delete;

  If DSDINGID='10001339' and
     DSDSUPID Not In('1000468300' '1000515600' '1000577200') Then Delete;

  If DSDINGID='10000920' and
     DSDSUPID Not In('1000618800' '1000671800') Then Delete;

  Keep DSDSUPID DSDINGID DSDINGR DSDQTY;
Run;
```

```
Data TEMP6(Rename=(NDSDQTY=DSDQTY));
   Set TEMP5;

                        *Calcium;
  If DSDINGID='10002190' and DSDSUPID='1000330700' Then Do;
                  DSDQTY=750;
                  NDSDQTY=DSDQTY*0.14; End;

  Else If DSDINGID='10000584' and DSDSUPID='1000317800' Then Do;
                        DSDQTY=5;
                        NDSDQTY=DSDQTY*0.0900; End;

  Else If DSDINGID='10002193' and DSDSUPID='1000330800' Then Do;
                        DSDQTY=147.7;
                        NDSDQTY=DSDQTY*0.236;
  End;

  Else If DSDINGID='10000795' and DSDSUPID='1000671600' Then Do;
                        DSDQTY=325;
                        NDSDQTY=DSDQTY*0.3;
  End;

  Else If DSDINGID='10004704' and DSDSUPID='1000589000' Then DSDQTY=500;

  Else If DSDINGID='10001144' Then
  Do;
        If DSDSUPID IN ('1000269000' '1000269001') Then DSDQTY=200;
        If DSDSUPID='1000320400' Then DSDQTY=30;
        NDSDQTY=DSDQTY*0.125;
  End;

  Else If DSDINGID='10001339' Then
  Do;
         DSDSUPID='1000468300' Then DSDQTY=140;
         If DSDSUPID='1000515600' Then DSDQTY=122;
         If DSDSUPID='1000577200' Then DSDQTY=125;
  End;

  Else If DSDINGID='10000920' Then
  Do;
        If DSDSUPID='1000618800' Then DSDQTY=10;
        If DSDSUPID='1000671800' Then DSDQTY=25;
        NDSDQTY=DSDQTY*0.1870;
  End;
                  *Iron;
  Else If DSDINGID='10000863' Then NDSDQTY=DSDQTY*.3300;
  Else If DSDINGID='10002217' Then NDSDQTY=DSDQTY*.1750;

                  *Lycopene;
  Else If DSDINGID='10004388' Then NDSDQTY=DSDQTY*.05;
  Else If DSDINGID='10001505' Then NDSDQTY=DSDQTY*.03;
  Else If DSDINGID='10006457' Then NDSDQTY=DSDQTY*.06;

                  *Lutein;
  Else If DSDINGID='10003229' and DSDSUPID='1000784200' Then NDSDQTY=1.12;
```

```
                  *Magnesium;
  Else If DSDINGID='10000612' Then NDSDQTY=DSDQTY*.4100;
  Else If DSDINGID='10000625' Then NDSDQTY=DSDQTY*.2890;
  Else If DSDINGID='10000585' Then NDSDQTY=DSDQTY*.0580;
  Else If DSDINGID='10002215' Then NDSDQTY=DSDQTY*.1830;

                  *Niacin;
  Else If DSDINGID='10000908' and DSDSUPID='1000146200' Then DSDQTY=.17;
  Else If DSDINGID='10001089' and DSDSUPID='1000458400' Then DSDQTY=320;

                  *Riboflavin;
  Else If DSDINGID='10001243' and DSDSUPID='1000438000' Then
  NDSDQTY=DSDQTY*.787;

                   *Thiamin;
  Else If DSDINGID='10000520' Then NDSDQTY=DSDQTY*.9200;
  Else If DSDINGID='10000904' Then NDSDQTY=DSDQTY*.8900;
  Else If DSDINGID='10000385' Then NDSDQTY=DSDQTY/40;

                    *Zinc;
  Else If DSDINGID='10001620' Then NDSDQTY=DSDQTY*.8034;
  Else If DSDINGID='10000586' Then NDSDQTY=DSDQTY*.1430;
  Else If DSDINGID='10000518' Then NDSDQTY=DSDQTY*.2500;

  Else NDSDQTY=DSDQTY;

  Drop DSDQTY;
Run;

Proc Sort Data=TEMP6;
  By DSDSUPID;
Run;

Proc Transpose Data=TEMP6 Out=TEMP7(Drop=_NAME_);
 Var DSDQTY;
 By DSDSUPID;
 ID DSDINGR;
Run;

Data Temp8(Rename=(Lutein_=Lutein));
  Set Temp7;

 Lutein_=Sum(of
 Lutein1,Lutein5,Lutein,Flowers,Marigold,Floraglo,Xeanthin,Zeaxanthin);
 Calcium = Sum (of Calcium1-Calcium10);
 DietaryFiber= Sum (of Dietary_Fiber1-Dietary_Fiber3);

 Drop Lutein Lutein1-Lutein5 Calcium1-Calcium10 Dietary_Fiber1-Dietary_Fiber3
      Flowers Floraglo Marigold Xeanthin Zeaxanthin;

Run;

Proc Sort Data=Temp8; By DSDSUPID;
Proc Sort Data=DSQV9; By DSDSUPID;
Run;
```

```
      /*** DSQV9 ----->  DSQV9_ ***/
Data DSQV9_;
  Merge DSQV9 (IN=A) Temp8 (IN=B);
   By DSDSUPID;
  If A;
Run;
      /*** DSQV9_ ----->  DSQV10 ***/
Data DSQV10;
   Set DSQV9_;
                        /* Calculate Ingredient Variables     */
                        /* Calculate Value by DSDACTSS*DSDQTY */
  DSQIKCAL=DSDACTSS*CALORIES;
  DSQIPROT=DSDACTSS*PROTEIN;
  DSQICARB=DSDACTSS*CARBOHYDRATE;
  DSQISUGR=DSDACTSS*SUGARS;
  DSQIFIBE=DSDACTSS*DIETARYFIBER;

  DSQITFAT=DSDACTSS*FAT;
  DSQISFAT=DSDACTSS*SATURATED_FAT;
  DSQIMFAT=DSDACTSS*MONOSATURATED_FAT;
  DSQIPFAT=DSDACTSS*POLYUNSATURATED_FAT;
  DSQICHOL=DSDACTSS*CHOLESTEROL;

  DSQIVB1=DSDACTSS*THIAMIN;
  DSQIVB2=DSDACTSS*RIBOFLAVIN;
  DSQIVB6=DSDACTSS*VITAMIN_B6;
  DSQIVB12=DSDACTSS*VITAMIN_B12;
  DSQIVC=DSDACTSS*VITAMIN_C;
  DSQIVK=DSDACTSS*VITAMIN_K;
  DSQIVD=DSDACTSS*VITAMIN_D;

  DSQILYCO=DSDACTSS*LYCOPENE;
  DSQILZ=DSDACTSS*LUTEIN;
  DSQINIAC=DSDACTSS*NIACIN;
  DSQICALC=DSDACTSS*CALCIUM;
  DSQIPHOS=DSDACTSS*PHOSPHOROUS;
  DSQIMAGN=DSDACTSS*MAGNESIUM;
  DSQIIRON=DSDACTSS*IRON;
  DSQIZINC=DSDACTSS*ZINC;
  DSQICOPP=DSDACTSS*COPPER;
  DSQISODI=DSDACTSS*SODIUM;
  DSQIPOTA=DSDACTSS*POTASSIUM;

  DSQIFA=DSDACTSS*FOLIC_ACID;
  DSQIFDFE=DSQIFA*1.7;

  DSQICHL=DSDACTSS*CHOLINE;
  DSQISELE=DSDACTSS*SELENIUM;
  DSQICAFF=DSDACTSS*CAFFEINE;

  If SEQN=43927 and DSDSUPID='1888084303' Then Do;
            DSDMTCH=5;
            DSQ128Q=.;
  End;
                  *Edits: DSDSUPID=1888028905, DSDMTCH=5, Done prior;
  If SEQN=51900 and DSDSUPID='1888028905' Then DSQ128Q=.;
  If SEQN=50661 and DSDSUPID='1888028905' Then DSQ128Q=.;
```

```
                  *Edit: DSDMTCH=5, Done prior;
  If SEQN=46309 and DSDSUPID='1888004301' and DSD090=91 Then Do;
            DSQ128A=.;
            Count=1;
  End;

  Else Do;
    DSQ128A=99;
    DSQ128R=.;
    Count=2;
 End;

                  *Manual Correction - Sort Order;
  If SEQN=49615 and DSDSUPID='1888036204' and DSQ128O=24 Then Count=1;
  Else If DSDSUPID='1888036204' and DSQ128O=. Then Count=2;

Drop CALORIES PROTEIN CARBOHYDRATE SUGARS DIETARYFIBER FAT SATURATED_FAT
     MONOSATURATED_FAT POLYUNSATURATED_FAT CHOLESTEROL THIAMIN RIBOFLAVIN
     VITAMIN_B6 VITAMIN_B12 VITAMIN_C VITAMIN_K VITAMIN_D LYCOPENE LUTEIN
     NIACIN CALCIUM PHOSPHOROUS MAGNESIUM IRON ZINC COPPER SODIUM POTASSIUM
     FOLIC_ACID
     CHOLINE SELENIUM CAFFEINE;
Run;
```